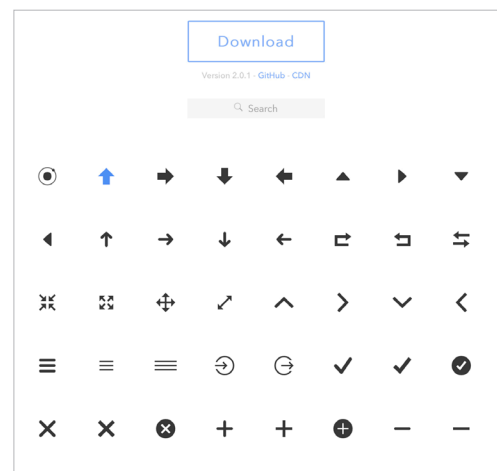
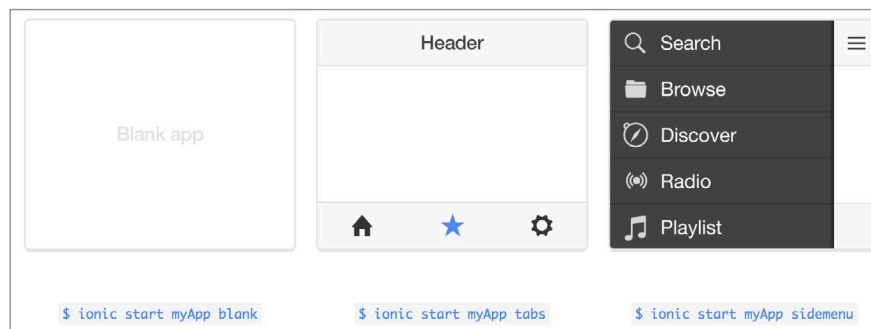


158 PC Professionale > Dicembre 2015

Un'applicazione si può creare a partire da un template, ecco i tre più comuni: app vuota, con interfaccia a schede e con menu laterale



un framework dedicato allo sviluppo web in generale.

Un secondo fronte, è la creazione di un ambiente di progettazione basato sul web, per creare applicazioni in modalità visuale, ancora in fase di sviluppo, ma molto interessante. Si tratta di uno strumento molto utile per la prototipazione rapida.

Un terzo fronte, forse il più interessante dal punto di vista strategico è una piattaforma di hosting di applicazioni, che permette di distribuire un'applicazione senza necessariamente inviarla ai negozi online di Apple e Google, avere una profilazione minuta delle funzioni che usano gli utenti, inviare notifiche push e forzare l'aggiornamento delle applicazioni. La piattaforma è ancora fase di test alfa, ma le implicazioni sono notevoli.

Un altro fronte di attività di Ionic è l'estensione del framework Angular, un'insieme di estensioni a html5 open source, realizzato da Google, per incorporare in modo più naturale i plugin di Cordova.

Certo, si tratta di obiettivi ambiziosi per una startup, ma su Crunchbase leggiamo che la società è forte di una quindicina di sviluppatori e ha ricevuto 2,6 milioni di dollari di finanziamento ad aprile 2015 per un totale di 3,7 milioni di dollari.

A maggio di quest'anno, Ionic ha concluso un accordo con IBM per fornire lo strumento di prototipazione rapida di applicazioni per la piattaforma MobileFirst della casa di Armonk (ibm.com/mobilefirst).

Bene, fin qui quello che riguarda la collocazione strategica di quello che stiamo per descrivere, ora entriamo nei dettagli tecnici, quelli che, come sviluppatori ci attirano di più.

L'AMBIENTE

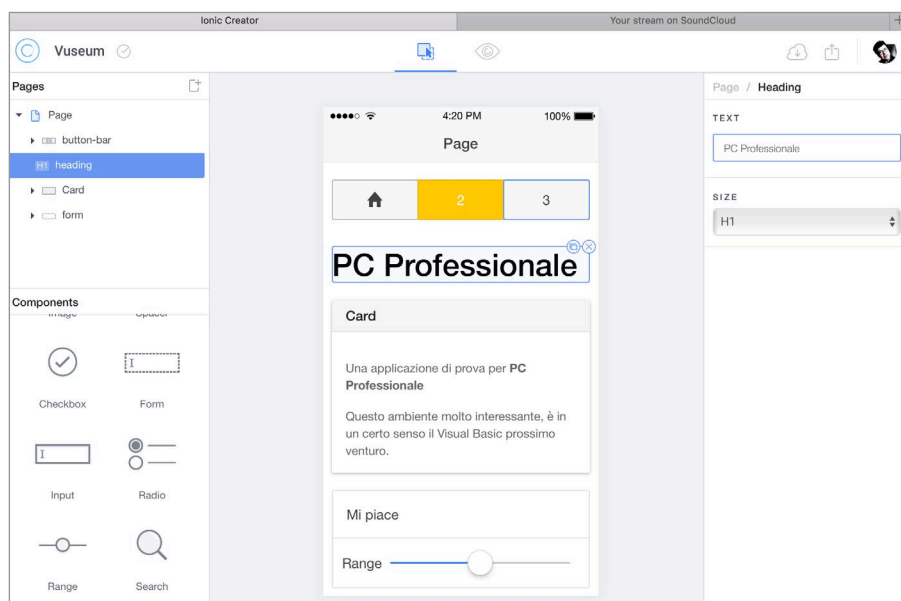
Come Phonegap, Ionic è costruito su Cordova, quindi si basa sull'ambiente run time Node.js. Di Node abbiamo parlato il mese scorso, descrivendo Phonegap. Ricordiamo che si tratta di un ambiente run time JavaScript basato sulla macchina virtuale v8 del browser Chrome, ricompilata come applicazione a sé stante.

Non solo Node (nodejs.org) fornisce un ambiente di esecuzione JavaScript slegato dal browser, ma consente la creazione di interfacce JavaScript per servizi nativi. Dato che il codice C di Node ha accesso completo alle funzioni della macchina, diventa possibile offrire l'accesso alle funzioni hardware dei telefoni, facendo da tramite fra il codice a basso livello e l'interprete JavaScript.

A Ionic sono associate un grande numero di icone, distribuite come font, con licenza open source MIT (ionicons.com)

Questo apre la strada alla creazione di applicazioni basate su html e JavaScript. Un'altra caratteristica notevole di Node è un sistema molto efficace per la pacchettizzazione, chiamato npm, che risolve il problema di tenere in ordine una libreria formata da componenti open source, spesso che hanno ritmi di aggiornamento rapidi, ma slegati fra loro e un intrico di dipendenze reciproche non facile da gestire.

Scaricare e installare Node è semplicissimo e le istruzioni sul sito di riferimento sono molto chiare.



Ionic Creator è un Ide per il toolkit Ionic, molto efficace per la progettazione rapida

LA RIGA DI COMANDO

Per installare l'ambiente Ionic, si procede da riga di comando con npm. Nel caso di Windows non occorrono precauzioni particolari, mentre su Linux o OS X, bisognerà lavorare in una shell con privilegio elevato (per esempio eseguendo `sudo sh`), oppure premettere `sudo` a tutti i comandi che alterano il sistema installando qualcosa. Ricordiamo che per avere accesso all'esecuzione privilegiata, dobbiamo aggiungere la nostra login al file `sudoers` (man `sudoers`, per informazioni). Annotiamo a margine che è un'ottima pratica quella di usare un utente senza poteri di amministrazione per il lavoro di tutti i giorni, una semplice assicurazione contro buona parte delle falle di sicurezza. L'installazione di Ionic e di Cordova, è molto semplice

```
[sudo] npm install -g cordova ionic
```

Quindi creiamo un progetto, per esempio da riga di comando, con

```
Ionic start MyApp tabs,
```

Su Windows abbiamo la possibilità di lavorare all'interno di Visual Studio Community, la versione gratuita, seguendo le indicazioni di questo tutorial taco.visualstudio.com/en-us/docs/tutorial-ionic. Ricordiamo che Taco è il nome di un progetto Microsoft per

incorporare in Visual Studio gli strumenti per lavorare con Cordova.

Nella riga di comando precedente, `MyApp` è il nome dell'applicazione e `tabs` è uno dei template disponibili, in questo caso un'applicazione con tre schede sovrapposte e una barra di pulsanti in basso per selezionare un'attività con tre icone predefinite, come `home`, `preferiti` e `impostazioni`. I template predefiniti sono `blank`, `tabs` e `side`. Quest'ultimo è un template con un menu laterale a scomparsa.

Ci sono diversi template non documentati sul sito primario, come `maps` e `salesforce`, per avere la lista in tempo reale conviene guardare la documentazione aggiornata della command line di ionic su [Github github.com/driftco/ionic-cli](https://github.com/driftco/ionic-cli)

Dopo avere creato un template, per eseguirlo nell'emulatore di iOS, bastano tre comandi:

```
[cd myApp]
ionic platform add ios
ionic build ios
ionic emulate ios
```

il primo comando aggiunge la piattaforma iOS al progetto, il secondo avvia la compilazione e il terzo fa partire l'emulatore. Naturalmente, tutto questo funziona se abbiamo fatto login su un Mac con Xcode installato. Naturalmente possiamo usare gli stessi comandi per creare applicazioni per Android o

qualche altro sistema operativo. La ragione numero uno per usare tecnologie basate su Cordova, infatti, è rispondere alla richiesta dei clienti di un'applicazione disponibile simultaneamente su più piattaforme.

Ionic offre la possibilità di testare l'applicazione in un browser, senza neanche ricompilare, esattamente come Cordova o Phonegap, ma offre l'opzione interessante di lanciare il server con due versioni dell'applicazione, costrette in un frame di dimensioni simili a quelle di uno schermo telefonico, in due versioni affiancate, una con l'aspetto dell'app su iOS e una su Android.

Le dimensioni dei frame sono approssimative e non rispecchiano necessariamente un device specifico, per esempio noi abbiamo misurato dimensioni uguali a 375 per 667 pixel. Questa modalità non sostituisce il test su un device, ma è efficace per farsi un'idea di dove si piazzano i diversi elementi di interfaccia, dato che device diversi hanno usanze diverse. Vedremo fra due sezioni che Ionic offre soluzioni molto più interessanti per provare un'app sul device.

IL FRAMEWORK

Il sito ionicframework.com presenta il framework applicativo che accompagna Ionic, che è costruito su Angular, una tecnologia creata da Google, estendendola e aumentandola con componenti visuali studiati allineati alle convenzioni delle piattaforme mobili.

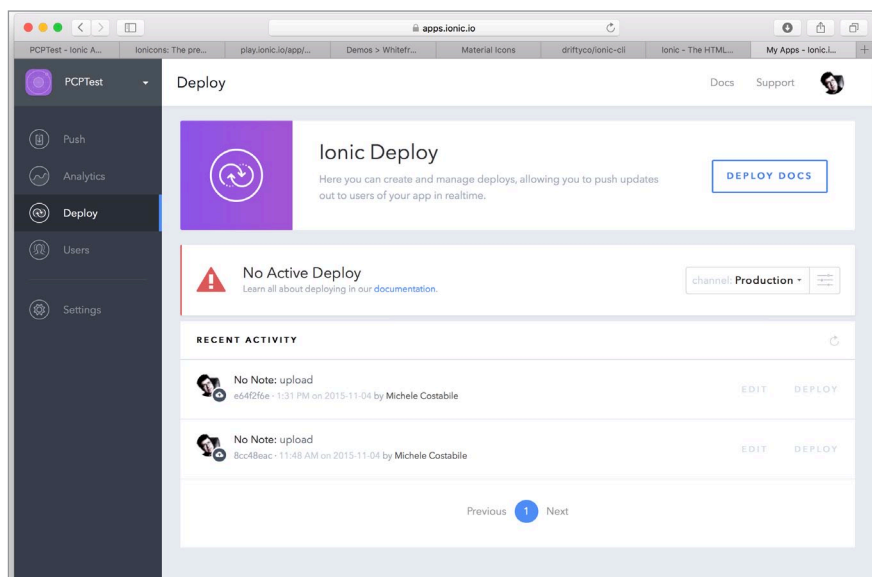
Per farci un'idea, non solo delle dimensioni del catalogo di componenti, ma anche della quantità di convenzioni su cui siamo abituati a contare nelle interfacce mobili, è utile fare un giro sul catalogo dei componenti visuali di Ionic ionicframework.com/docs/components.

Il framework comprende anche una notevole quantità di icone, tutte di buona qualità, espressive, familiari, vettoriali e scalabili, in una parola utilizzabili. Le icone sono raccolte in un font distribuito con licenza open source MIT e disponibili sul sito ionicons.com.

Nello stile di Angular, il codice html viene arricchito di direttive e decorato con stili studiati accuratamente per dare l'effetto di componenti nativi.

Ecco un esempio di codice:

```
<body ng-app="app">
  <ion-pane>
    <ion-header-bar>
```



La dashboard con le applicazioni ionic che abbiamo creato su ionic.io dà molte informazioni, le statistiche di uso dettagliate, l'elenco degli utenti e la possibilità di inviare notifiche push.

```

class="bar-energized">
  <h1 class="title">PC
  Professionale</h1>
</ion-header-bar>
  <ion-content class="padding">
    <ul class="list">
      <li class="item">
        Raccogliere il
materiale
      </li>
      <li class="item
assertive">
        Verificare le fonti
      </li>
      <li class="item">
        Creare un prototipo
      </li>
      <li class="item">
        Scrivere l'articolo'
      </li>
    </ul>
    <button class="button
button-balanced">Inviarlo!</button>
  </ion-content>
</ion-pane>
</body>

```

Questo codice mostra una pagina con un banner di colore giallo, una lista di elementi, di cui uno evidenziato, e un pulsante verde, come vediamo nella figura.

Osserviamo nel codice come le liste sono decorate con una classe, i colori sono rappresentati da stati d'animo e nel codice html si mischiano elementi nativi con elementi che non fanno parte di html, come ion-content e ion-pane. Come fa Angular, Ionic estende html aggiungendo elementi che vengono utilizzati dal codice JavaScript per essere infine tradotti negli elementi appropriati, con il dovuto set di stili applicati, il comportamento dinamico e il binding dei dati.

In poche parole, Angular e quindi Ionic estendono html per farlo somigliare un po' a Xaml, il linguaggio di definizione di interfacce utente che dovrebbe essere ben familiare agli sviluppatori Microsoft.

Si tratta di un approccio che condividiamo, perché in questo modo i template conservano un aspetto riconoscibile e possono essere scambiati fra sviluppatori e grafici in modo costruttivo. Altri approcci basati esclusivamente sul codice sono accessibili solo a chi programma e rallentano la comunicazione nel team.

Notiamo, però, che il markup di un'applicazione ionic si arricchisce di elementi del tutto proprietari, quindi comporta un legame con la piattaforma che va valutato strategicamente.

Ionic.io

Il sito Ionic.io ospita la piattaforma Ionic, che è ancora in alfa test, ed è uno dei tasselli più interessanti del progetto. La piattaforma offre diversi servizi agli sviluppatori: hosting, compilazione, prototipazione rapida e distribuzione. Un account sul sito ci consente di ospitare le nostre applicazioni, per compilarle, condividerle analizzarle e distribuirle in modi interessanti e non convenzionali. La riga di comando permette di inviare un'applicazione al sito, per esempio

ionic upload

da dentro la cartella dell'applicazione. Dopo avere fatto upload dell'applicazione possiamo richiederne la compilazione, ma il servizio più interessante lo dà la Ionic Vire app, che si trova sugli store Apple e Google. L'app è un host generico che permette di provare un'applicazione senza compilarla, esattamente come l'applicazione analoga di Phonegap, ma il valore aggiunto interessante sta nel fatto che la app in prova può essere distribuita tramite il sito di Ionic, quindi è possibile compilare una applicazione e distribuirla a un insieme di clienti di prova senza doverla

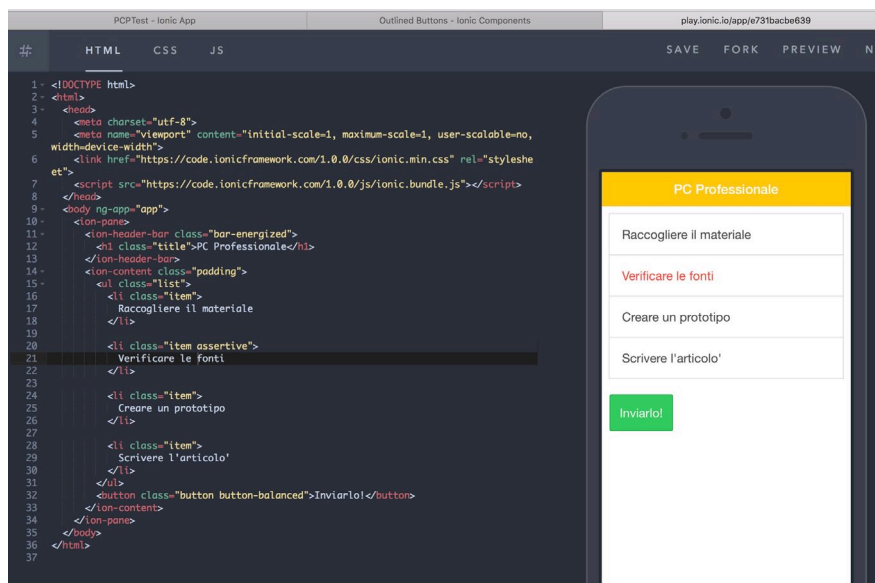
ricompilare e sottomettere agli store. Questa possibilità è molto interessante nel caso che si voglia condividere con i clienti il progresso dello sviluppo di una app, o che si vogliano distribuire a un pubblico selezionato app per uso interno, per esempio un catalogo per i venditori o altre applicazioni di interesse aziendale.

Il cruscotto di un'applicazione Ionic permette di inviare notifiche push, di aggiornare gli utenti dividendoli per canali, per esempio discriminando fra un canale ufficiale, uno sperimentale e uno di alfa test e, infine, si possono osservare statistiche d'uso raccolte dall'applicazione. In questo modo, si possono monitorare gli utenti e il loro modo di usare l'applicazione, per raccogliere informazioni utili per il ridisegno dell'interfaccia, ad esempio.

Queste funzionalità caratterizzano molto la posizione di mercato di Ionic, non solo si tratta di una piattaforma di sviluppo, ma anche di distribuzione e di test delle applicazioni, che semplifica grandemente la produzione di determinate app aziendali per uso interno o che hanno un pubblico limitato e non richiedono per forza la pubblicazione tramite l'AppStore, che impone vincoli e tempi di attesa indicati per le applicazioni di uso generale.

IONIC CREATOR

Un altro dei contributi interessanti di Ionic è Creator (creator.ionic.io)



Su playground.ionic.io troviamo uno strumento per il test rapido online più curato esteticamente di altri progetti simili, come Codepen, ma ancora immaturo.

un'ambiente per il disegno visuale di applicazioni ospitato nel browser. Con Creator si può creare l'interfaccia utente di un'applicazione in modo visuale, utilizzando una ricca tavolozza di componenti, in modo rapido, come siamo abituati a fare sul desktop. L'applicazione creata sul sito può essere scaricata in locale con il comando

```
ionic start [appName]
creator:ac97c32e8f4
```

usando creator, seguito da due punti e dall'identificativo di un'applicazione come nome del template da utilizzare per la creazione. Sul computer possiamo aggiungere il codice necessario per animare l'interfaccia e, quando siamo pronti per la distribuzione, possiamo fare upload sulla piattaforma, come abbiamo mostrato poco sopra. Ionic ospita anche un ambiente per prove veloci, all'indirizzo play.ionic.io. Si tratta di una variazione sul tema degli editor online, fra cui possiamo citare codepen.io, jsbin.com e plnkr.co. Il playground di Ionic è carino, ma non offre niente di più della concorrenza, casomai di meno, perché non ci è riuscito di salvare un progetto per riprenderlo in seguito. Dato che la visualizzazione ha lo aspect ratio del video di un telefono, può capitare che torniamo su questo playground per una prova veloce, ma nulla di più, almeno fino a che non è completata la probabile integrazione con l'area di lavoro, i progetti attivi e una funzione save funzionante.

ngCordova

L'ultimo contributo di Ionic allo sviluppo di applicazioni Cordova è un progetto collaterale, ma utile, chiamato ngCordova, che offre una collezione di oltre 70 plugin Cordova, pacchettizzati come estensione Angular. Il beneficio di ngCordova sta nel modo in cui i plugin si possono utilizzare. Per esempio, il modo in cui si attiva la macchina fotografica con Cordova è

```
navigator.camera.
getPicture(cameraSuccess,
cameraError, [ cameraOptions ]);
```

in cui cameraSuccess e cameraError sono funzioni callback.

Con ngCordova, la chiamata diventa

```
$cordovaCamera.getPicture(options).
then(function(imageData) {
// Codice da eseguire in caso
positivo
}, function(error) {
// Codice da eseguire in caso
di errore
});
```

In entrambi i casi, si attacca del codice a un evento futuro, nel caso della callback in modo esplicito, nel caso di una promise, come nella funzione restituita da getPicture, abbiamo un oggetto che ha un metodo .then che viene invocato nel momento stabilito con la funzione, in questo caso anonima, che viene invocata.

Le promise sono un costrutto sintattico, su cui dovremmo spendere qualche parola in più, ma per il momento, ci basta dire che si tratta di oggetti che rappresentano qualcosa che non è ancora disponibile a cui collegare del codice che sarà eseguito quando quel qualcosa è diventato disponibile. Sono un modo semplice e elegante di gestire l'asincronicità del web, fatto di server che rispondono in tempi non sempre prevedibili. In questo caso, la necessità di usare codice asincrono sta nel fatto che Cordova si basa su Node.js, che è un'applicazione single threaded, e non si può tenere bloccata l'intera applicazione fino a quando l'apertura della macchina fotografica è completata.

Le promise diventano interessanti quando si collegano in serie più processi potenzialmente asincroni, come in questo codice:

```
$unServizio.dammiUnData()
.then(function(result) {
// Fai qualcosa
})
.then(function(result){
// Fai altro
})
.then(function(result){
// E infine ...
});
```

CONCLUSIONI

Ionic ci è piaciuto molto, ma alla domanda se lo sceglieremmo per un prossimo progetto mobile, la risposta è "dipende". Si tratta di una startup

in fase di decollo, ma le persone e i finanziamenti ci sono, così come ci sono i risultati e le collaborazioni. Quindi, anche se il progetto non è ancora maturo, si può aspettare che lo diventi. Di conseguenza, se l'orizzonte temporale non è troppo vicino, ci si può investire. L'ambiente interattivo di prototipazione rapida e disegno, la possibilità di condividere l'applicazione in tempi brevi, anche senza passare dall'AppStore, sono tutte funzioni molto interessanti quando diventano un fattore chiave per tenere insieme un team distribuito. Si può mandare un feedback immediato ai clienti o ai destinatari di un'applicazione e

aprire le porte del laboratorio ai clienti, senza organizzare meeting continui.

La possibilità di tracciare l'uso dell'applicazione e raccogliere dati di uso sul campo è molto interessante e può essere preziosa nel caso di prototipi.

Un'altra possibilità molto interessante è quella di automatizzare gli aggiornamenti, sia inviando notifiche push agli utenti dal cruscotto di Ionic.io, sia automatizzando l'aggiornamento nel codice dell'applicazione usando il supporto del framework. I creatori di Ionic hanno detto esplicitamente di essere interessati a quell'universo di applicazioni che non sono destinate al grande pubblico, per esempio programmi per la forza vendita o per gli operatori mobili di una struttura pubblica, applicazioni in cui è importante non dover sottostare alle politiche di sottomissione dell'AppStore e poter fare aggiornamenti con una frequenza elevata, quando serve anche in giornata.

Nelle nostre prove siamo riusciti a creare un'app per Android e iOS e inviarla ai rispettivi dispositivi senza nemmeno utilizzare una licenza developer per l'AppStore, un'altra possibilità interessante.

Se parliamo di sviluppo generico di un'applicazione, o magari di un gioco, può darsi che entrare nei meandri del framework sia inutilmente gravoso, che la struttura dei componenti, molto codificata, ponga eccessive rigidità, che rimanere sul semplice trasportando quasi senza modifiche un'applicazione web già sviluppata sia la soluzione ideale per chi parte da un asset web.

Senza licenza

È possibile collaudare l'app in fase di sviluppo direttamente sui dispositivi anche senza licenza developer